

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

КУРСОВА РОБОТА

з дисципліни “Бази даних”

спеціальність 121 – Програмна інженерія

на тему: «**Моніторингова система мобільних додатків у Play Store**»

Студентка групи КП-91

**Чорна Софія
Олександрівна**

(підпис)

**Викладач
к.т.н, доцент кафедри СПіСКС**

Петрашенко А.В.

(підпис)

Захищено з оцінкою _____

Київ – 2020

Анотація

Курсова робота є моніторинговою системою мобільних додатків у Play Store, що має консольний інтерфейс для взаємодії із базою даних, а також можливість провести аналіз даних та вивести його у графічний файл.

У результаті виконання курсової роботи були набуті практичні навички розробки сучасного програмного забезпечення, що взаємодіє з реляційною базою даних. Також були здобуті навички оформлення відповідного текстового, програмного та ілюстративного матеріалу у формі проектної документації.

Окрім цього було здобуті навички володіння основами СУБД, а також інструментальними засобами розробки додатків для подібних баз даних.

Зміст

Вступ	5
Аналіз інструментарію для виконання курсової роботи	6
Структура бази даних	7
Опис програмного забезпечення	7
Загальна структура програмного забезпечення	7
Опис модулів програмного забезпечення	8
Опис основних алгоритмів роботи	9
Аналіз функціонування засобів реплікації	9
Аналіз функціонування засобів резервування/відновлення	9
Аналіз результатів підвищення швидкодії запитів	9
Опис результатів аналізу предметної галузі	10
Висновки	11
Література	12
Додатки	13
Графічні матеріали	13
Фрагменти програмного коду	20

Вступ

Сьогодні існує декілька популярних онлайн-платформ, які дозволяють стороннім компаніям пропонувати власникам телефонів встановлювати і купувати різні додатки. Це App Store, Google Play та інші.

Ринок мобільних додатків дуже розвинений і неухильно зростає. Згідно з прогнозами Statista, в 2020 році валовий річний дохід в галузі мобільних додатків перевищить \$ 189 млрд. Тому було вирішено обрати саме цю тему для курсовою роботи.

З метою оцінки певних трендів ринку Android створена система може відслідковувати залежності між категоріями, типом додатків, кількістю завантажень.

Аналіз інструментарію для виконання курсової роботи

Для виконання курсової роботи було обрано наступні інструменти:

База даних — PostgreSQL. У даній СУБД підтримується нормалізована форма збереження даних, об'єднання даних, зовнішні ключі, індекси, реплікація тощо. До того ж, PostgreSQL є безкоштовною та має потужну мову запитів. Серед недоліків — відносно складна реалізація резервування.

Бібліотека для роботи із базою даних — psycopg2. Найпопулярніший адаптер бази даних PostgreSQL для мови програмування Python.

Бібліотека для візуалізації різних математичних даних — matplotlib. Вона має прості для розуміння людиною високорівневі функції у модулі matplotlib.pyplot. На сайті бібліотеки наявна вичерпна документація, а також детальні туторіали з побудовою різних типів діаграм. Проте виникли складнощі з відрегулюванням елементів графіків відносно один одного.

Фреймворк для скрапінгу веб-сайтів — Scrapy. Фреймворк дозволяє відносно швидко та ефективно парсити велику за обсягом роботу. Проте, навчання тому, як він працює, вимагає багато часу, але після освоєння процес сканування перетворюється в один рядок коду.

Код курсової роботи був написаний мовою програмування Python 3 у середовищі PyCharm Community Edition 2020.

Структура бази даних

База даних має наступні таблиці:

- apps – містить інформацію про мобільні додатки;
- apps_types – містить інформацію про тип оплати додатку;
- app_categories – містить інформацію про категорію додатку;
- developers – містить інформацію про розробників мобільних додатків;
- dev_app_links – містить інформацію про зв'язок між розробниками, мобільними додатками та датою їхнього випуску;
- users – містить інформацію про користувачів Play Store;
- ratings – містить інформацію про відгуки користувачів на додаток.

Схема бази даних міститься у Додатку 1.

Опис програмного забезпечення

Загальна структура програмного забезпечення

Програмні засоби містять наступні компоненти:

1. Підсистема попередньої обробки даних, що складається з:
 - 1.1. Засоби генерації даних. Забезпечена адекватна генерація даних за допомогою датасетів та парсингу сторінок додатків у Play Store.
 - 1.2. Засоби фільтрації та валідації даних.
2. База даних системи призначена для зберігання, аналізу та реплікації інформації розробленої моніторингової системи.
3. Засоби реплікації, що входять у склад сервера бази даних і призначені для забезпечення цілісності, узгодженості та доступності даних, що зберігаються у СУБД.
4. Засоби аналізу даних.
5. Засоби резервування та відновлення даних, що призначені станів бази даних у різні моменти часу.

6. Засоби атрибутивного та повнотекстового пошуку.

Опис модулів програмного забезпечення

Модуль model:

Взаємодіє із базою даних, містить усі запити для отримання, вилучення, вставки або редагування даних. Для його реалізації була використана бібліотека psycorp2.

Модуль view:

Приймає введені користувачем дані, які передає модулю контролера, демонструє результати виконання відповідних обраних дій.

Модуль controller:

Допомагає взаємодіяти модулю model та модулю view між собою. При відповіді модуля model він форматує дані для зручного їх відображення у view.

Модуль google_play_scapper:

Парсить сторінки мобільних додатків з Play Store. Містить широкий функціонал для отримання об'єктів app та review за параметрами мови коментарів, кількості об'єктів, поставлених оцінок, країною. Взаємодіє з модулем csv_parse.

Модуль generation:

Взаємодіє з модулем google_play_scapper, csv_parse та базою даних. Було обрано 10 найпопулярніших додатків з категорії “Продуктивність” у Play Store, які передаються у google_play_scapper для парсингу. Таким чином є можливість згенерувати достатню кількість даних. Також для генерації даних використовуються датасети.

Модуль visualization:

Створює, демонструє та зберігає різні види графіків.

Опис основних алгоритмів роботи

Реалізовано функції для знаходження найпопулярніших додатків, категорій, розробників, аналізу дат релізів та типів мобільних додатків. Отримані дані були використані у візуалізації. Результати містяться у Додатку 2.

Також реалізовано пошук по схожим назвам додатків та вмісту коментарів у відгуках. Було створено тригери для підтримання стовпців `vs_vector`, що використовуються для пошуку, у таблицях `apps` та `ratings` в актуальному стані.

Аналіз функціонування засобів реплікації

Реплікація була реалізована за принципом `master-slave`, шляхом використання сервера на віртуальній машині як `slave`-сервера. Створено додаткового користувача з правами на реплікацію. При несправності `master`-сервері на основній машині, роль `master` передається `slave`.

Аналіз функціонування засобів резервування/відновлення

За планом було реалізувати диференційне резервне, але СУБД має повну підтримку лише повного резервного копіювання. Воно було реалізовано за допомогою програмного забезпечення `SQLBackupAndFTP`, яке дозволяє просто та швидко налаштувати заплановане резервне копіювання, обирати бази даних та місце зберігання копій, сжимати їх у `zip`-формат. Також `SQLBackupAndFTP` повідомляє про можливі помилки та їх причини. Резервне копіювання заплановано о 19:30 кожного дня.

Аналіз результатів підвищення швидкодії запитів

Для підвищення швидкодії виконання запитів було створено індекси. Для повнотекстового пошуку таблицях `apps` та `ratings` створено стовпці типу `ts_vector`, які автоматично оновлюються при зміні цих полів. Це мало ряд переваг зберігання обчисленого виразу індексу в окремому стовпці. По-перше, для використання індексу в запитах не потрібно явно вказувати ім'я конфігурації текстового пошуку. По-друге, пошук виконується швидше, так як для перевірки відповідності даних індексу не потрібно повторно виконувати `to_tsvector`. Це дозволяє не генерувати дані вектори

при пошуку, а лише при його зміні. Також для цього вектора використовується ще й індекс типу GIN, що робить повнотекстовий пошук більш швидким.

Також був використаний індекс BRIN для стовпця released_date у таблиці dev_app_links. Перед створенням було перевірено кореляцію. Стовпець released_date має значення кореляції біля одиниці, а отже доречно використовувати індекс.

Час запитів з використанням створених індексів помітно зменшується, що видно на графіку у Додатку 3.

Опис результатів аналізу предметної галузі

Для аналізу мобільних додатків було проаналізовано кількість завантажень, адже від їхньої кількості залежить популярність. За результатами аналізу більш ніж 17000 додатків найпопулярнішим виявлено Google. На категорію 'Tools' припадає найбільша кількість завантажень, за нею йде категорія 'Communication'. Природньо, що рейтинг розробників очолює Google, на другому місці - Skype, та замикає трійку лідерів Opera.

Найрозповсюдженіший тип додатків - free, він складає близько 60%. Натомість partially free становить 36.1%. Повністю платні (paid) додатки складають лише 3.8% з проаналізованих.

Також було розглянуто дати релізів. Починаючи з 2011 року, кількість додавань розробниками нових додатків у Play Store постійно росла. Так, на кінець 2020 року маємо найбільшу кількість релізів саме у 2020 році. Графіки з проаналізованими даними знаходяться у Додатку 4.

Висновки

На даній курсовій роботі було розроблено додаток для аналізу магазину додатків, а також ігор, книг, музики і фільмів від компанії Google - Play Store.

Для збору інформації про додатки, категорії, типи, користувачів, відгуки, розробників було використано веб-сторінки Play Store 10 найпопулярніших додатків з категорії 'Productivity'. Зібрана інформація використовується для генерації даних до таблиць ratings та users. Крім того, може бути використана для генерації й інших таблиць. Було використано датасети для генерації apps та developers.

Для резервування та відновлення було використано вже готову утиліту із графічним інтерфейсом SQLBackupAndFTP, яка забезпечує легке автоматичне та ручне резервування та відновлення, за потреби та за планом.

Налагоджено реплікацію між master-сервером та slave-сервером, що знаходиться на віртуальній машині. В разі неполадок на основному сервері, запити на зчитування інформації перенаправляються на slave-сервер.

Для пришвидшення повнотекстового пошуку пошуку були введені індекси та поля із ts_vector розбором цього тексту.

Засоби аналізу даних дозволяють створити графічні елементи на основі проаналізованих даних. Так, було відслідковано динаміку релізів, популярність додатків, категорій, розробників та типи додатків.

Засоби валідації та фільтрації даних використовуються для контролю коректних вхідних даних від користувача.

Для підвищення швидкодії звернень до таблиці apps та ratings було додано індекси GIN на стовпці, що містять tsv_vectors. Також було створено BRIN індекс на стовпець 'released_date' з таблиці зв'язків dev_app_links.

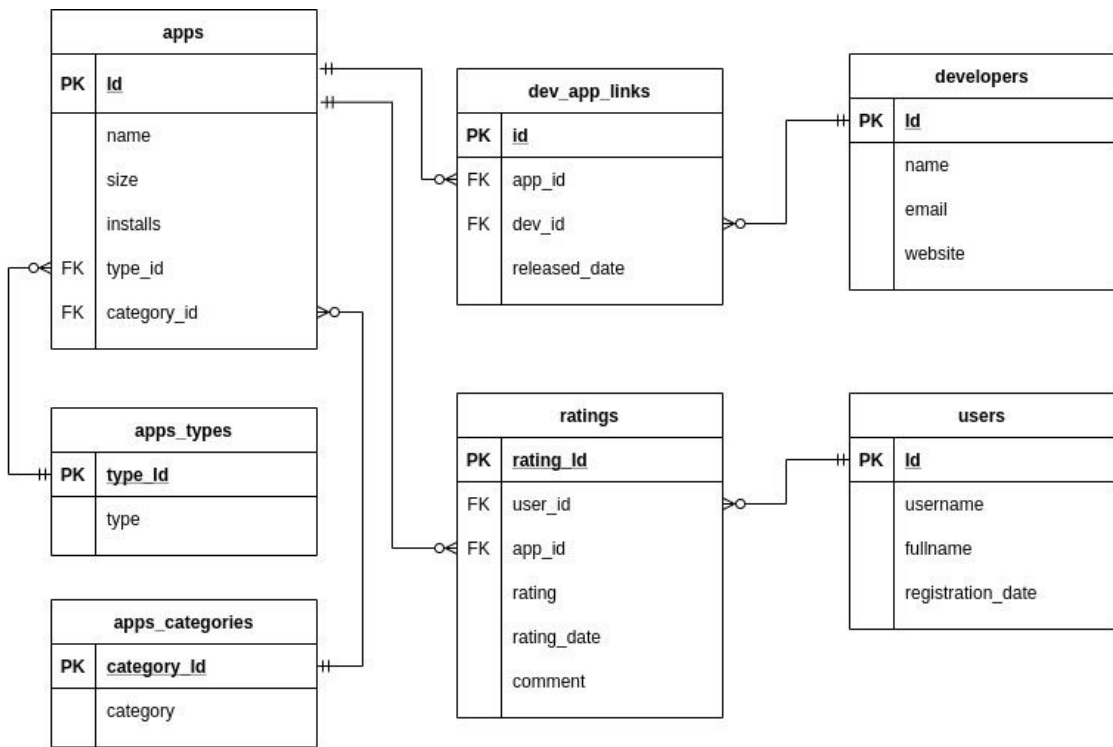
Отже, мета даної курсової роботи була досягнута.

Література

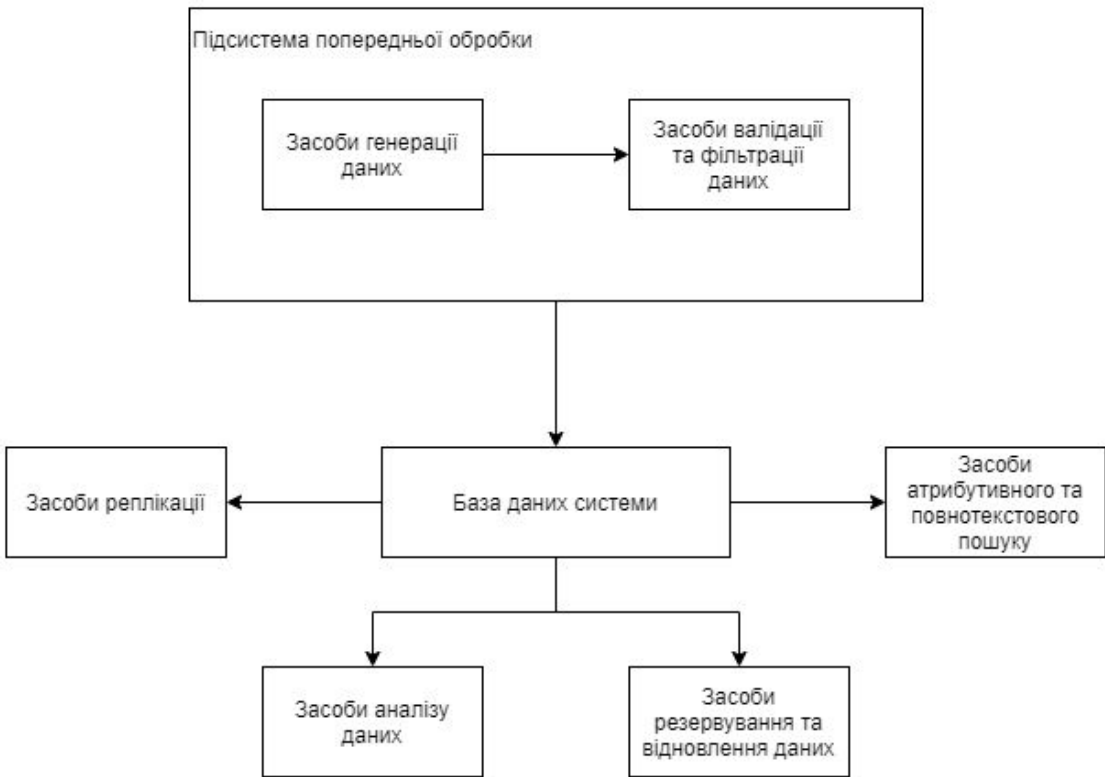
1. What is PostgreSQL? <https://www.educba.com/what-is-postgresql/>
2. TOP 5 PYTHON LIBRARIES FOR VISUALIZATION
<https://www.techshenanigans.com/post/top-5-python-libraries-for-visualization>
3. Индексы в PostgreSQL
<https://habr.com/ru/company/postgrespro/blog/346460/>
4. Отказоустойчивый кластер Master-Slave на PostgreSQL
<https://habr.com/ru/post/188096/>
5. Matplotlib: Visualization with Python <https://matplotlib.org/>
6. Scapy documentation <https://scapy.org/>

Додатки

Графічні матеріали













Структура бази даних



Структура програмних засобів

Job Settings

 Server	 SW
 PostgreSQL Server	localhost:5432
 Selected databases	play_store
 Destinations	 Folder (home/sofiyachernaya/Postgresql/ba...
 Schedule backup	Full: every 24 hr next start: 7:30 PM
 Compression	.zip
 Restore	Restore to  SW PostgreSQL Server: localhost:5432 play_store → play_store_2

SQLBackupAndFTP

General information

Backup job: **Backup job 1**

Server:  SW

DBMS: PostgreSQL Server: localhost:5432

Status: **Success**

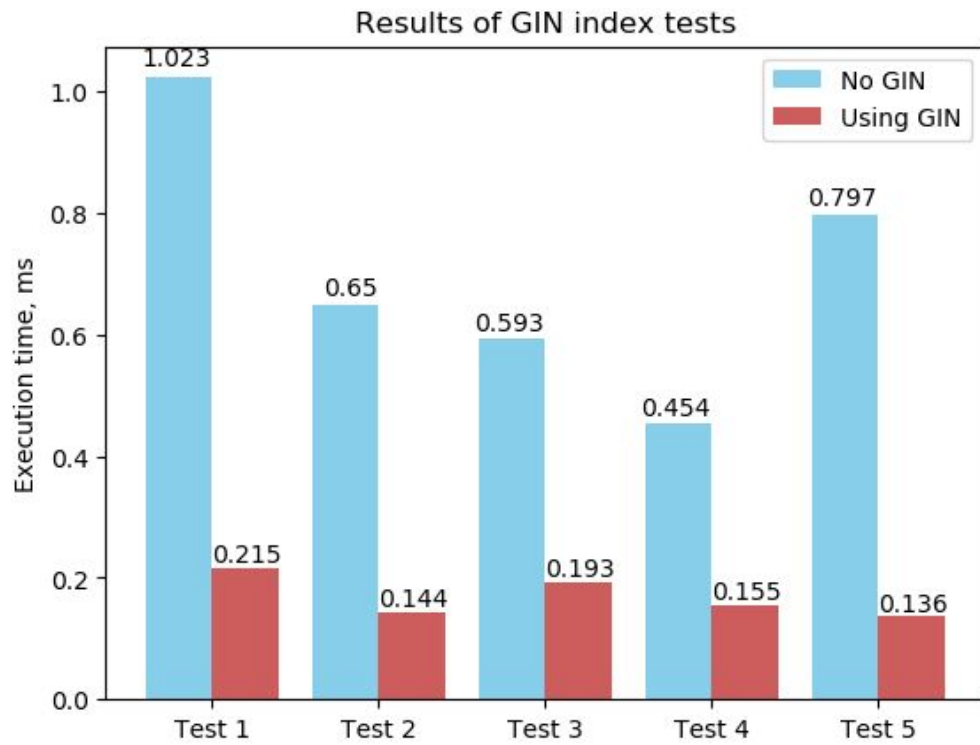
Start time: 12/16/2020 12:42:02 PM

Duration: 00:00:53

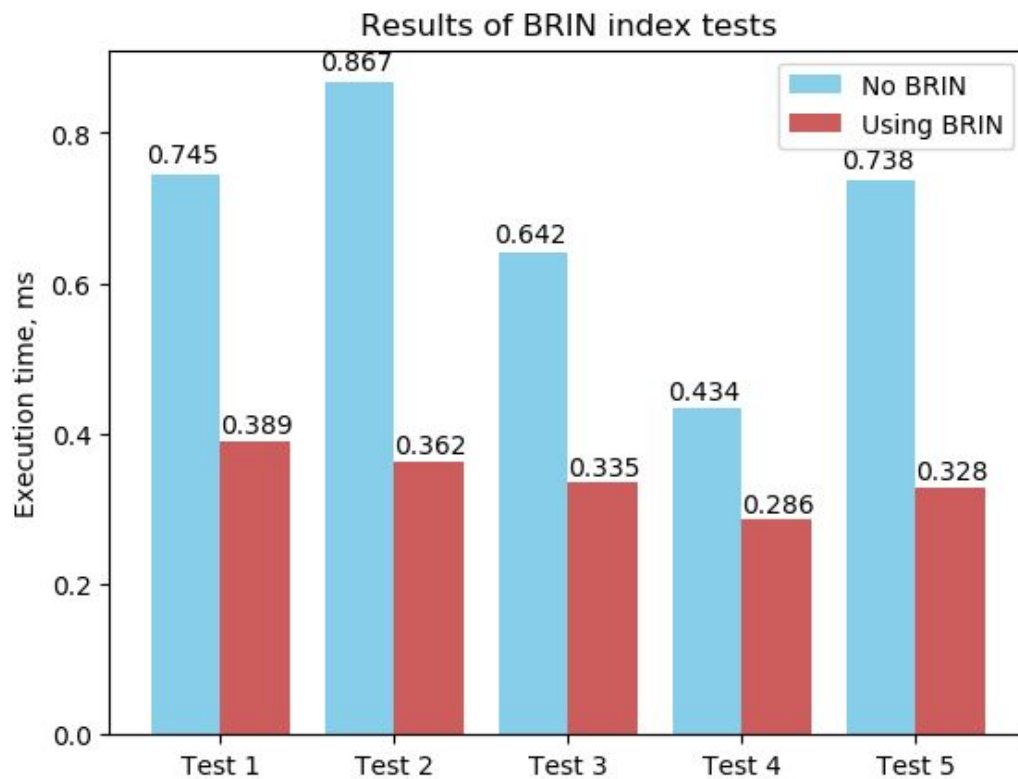
Size: 1018.21 KB

Archive size: 378.2 KB

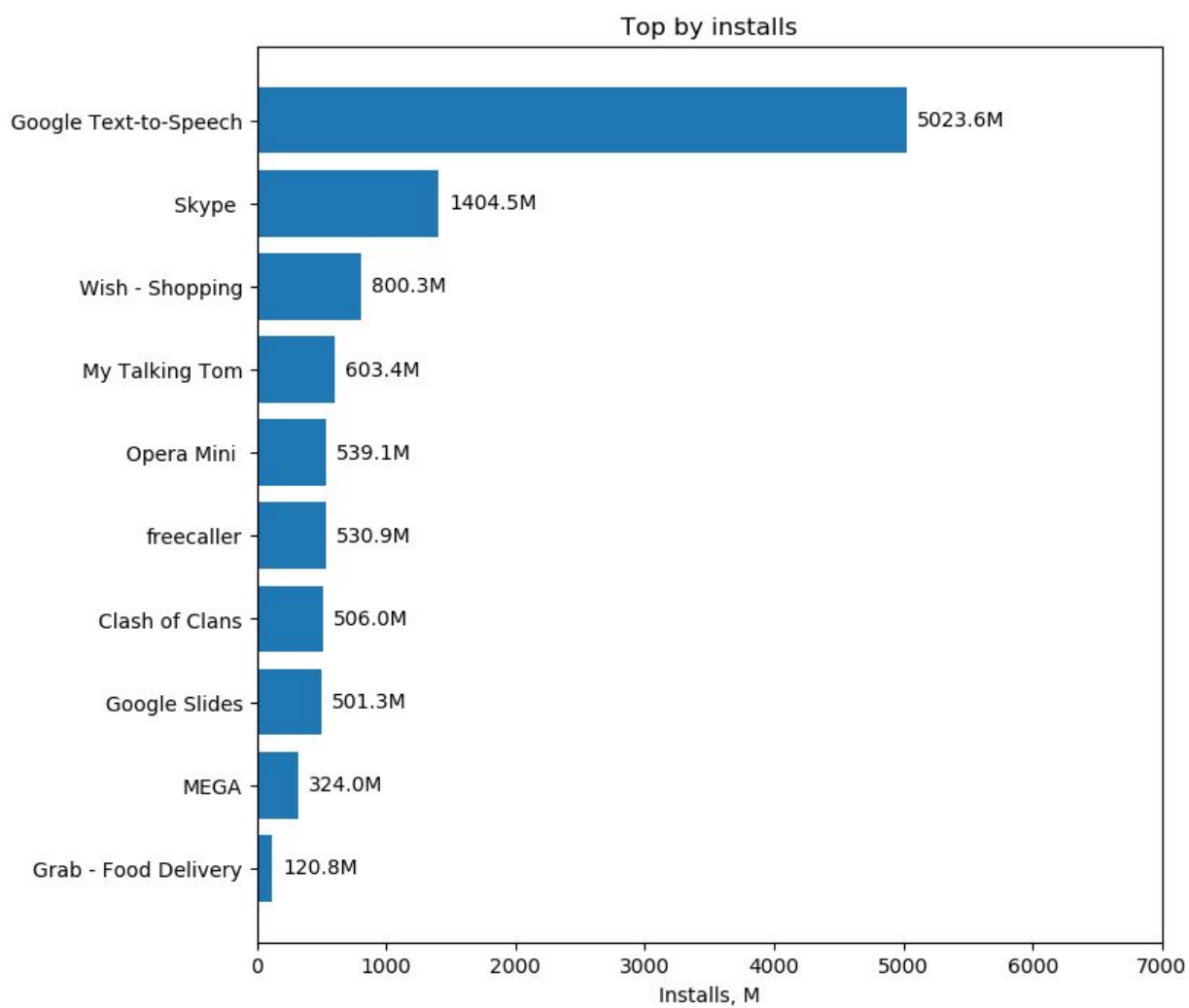
Час резервування та розмір збережених даних



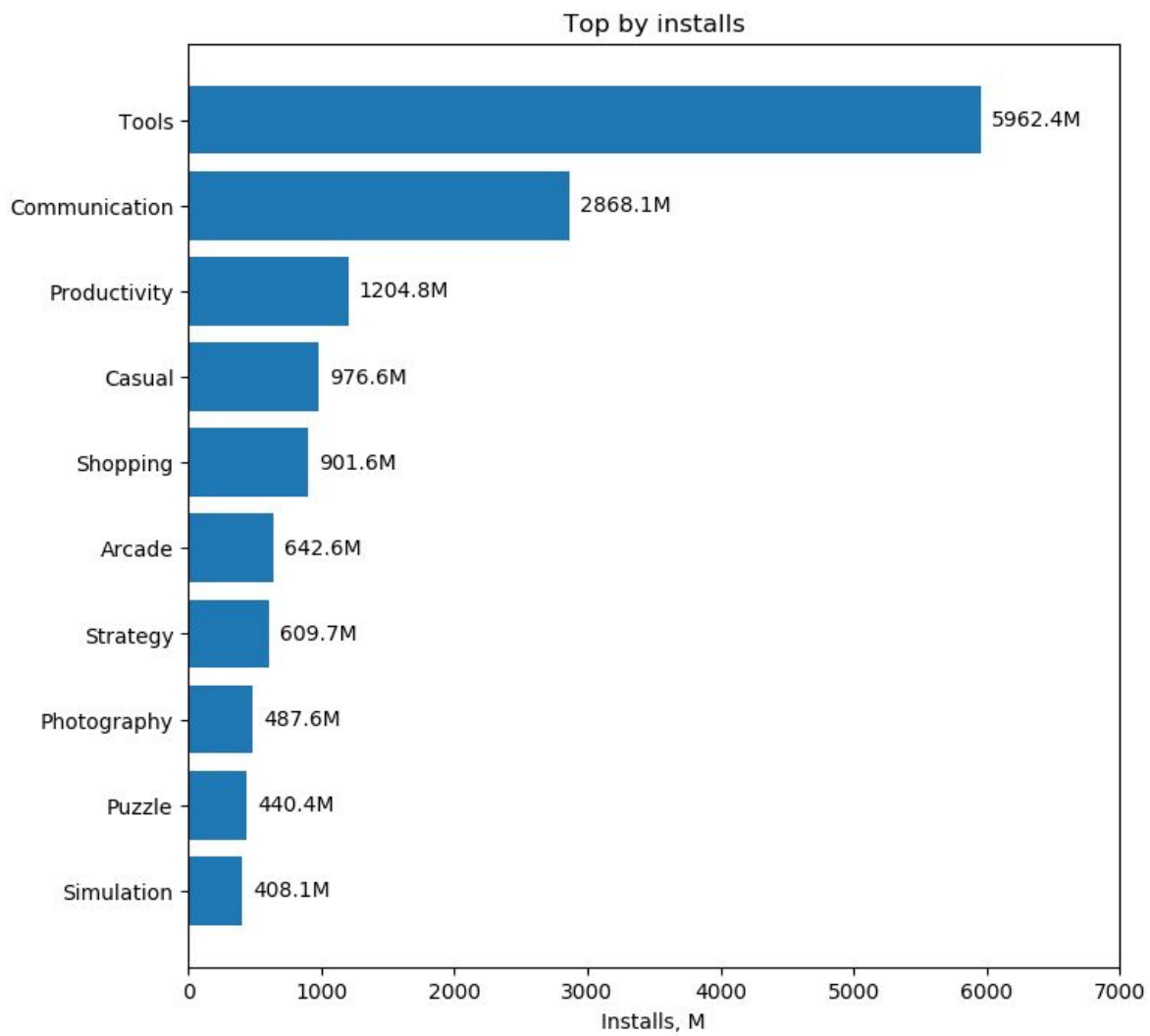
Результати тестів індексу GIN



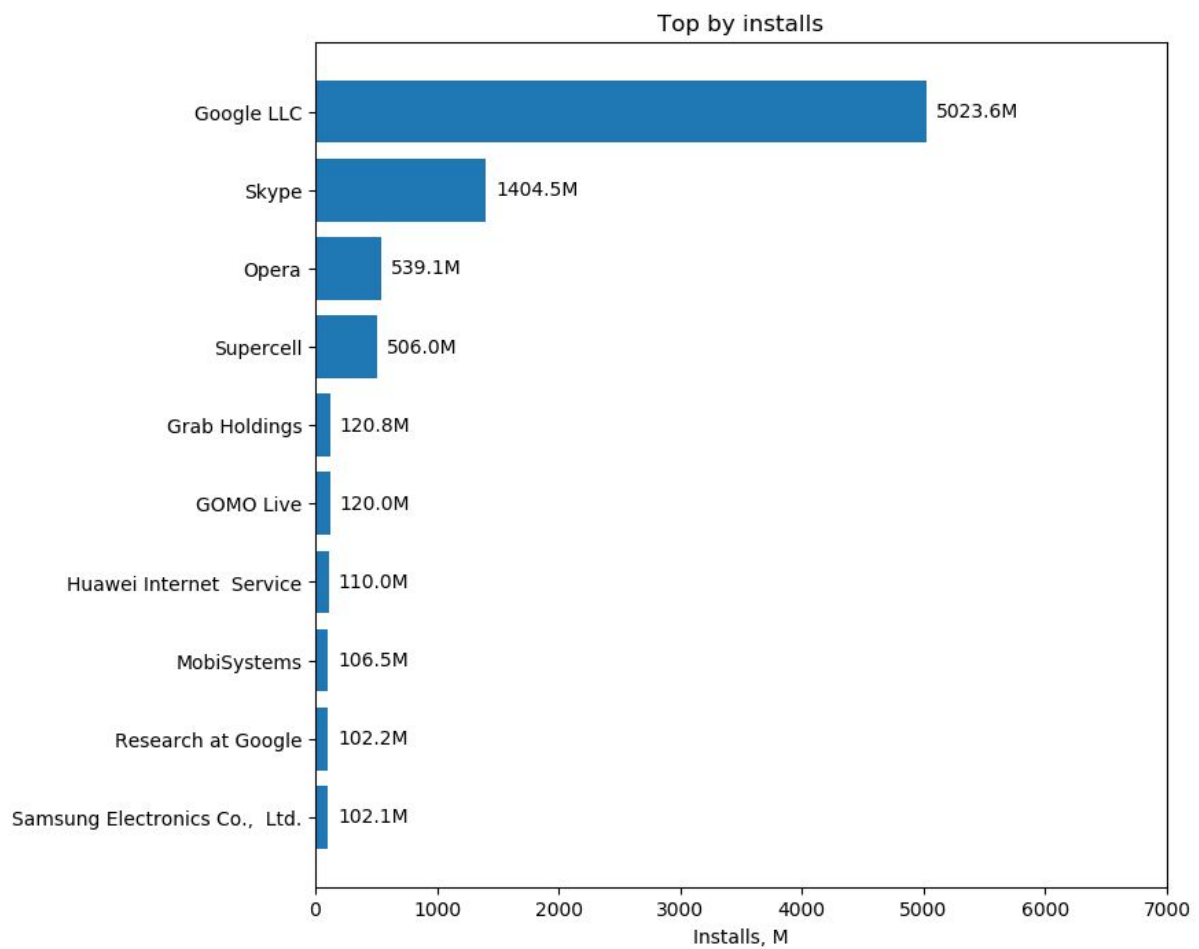
Результати тестів індексу BRIN



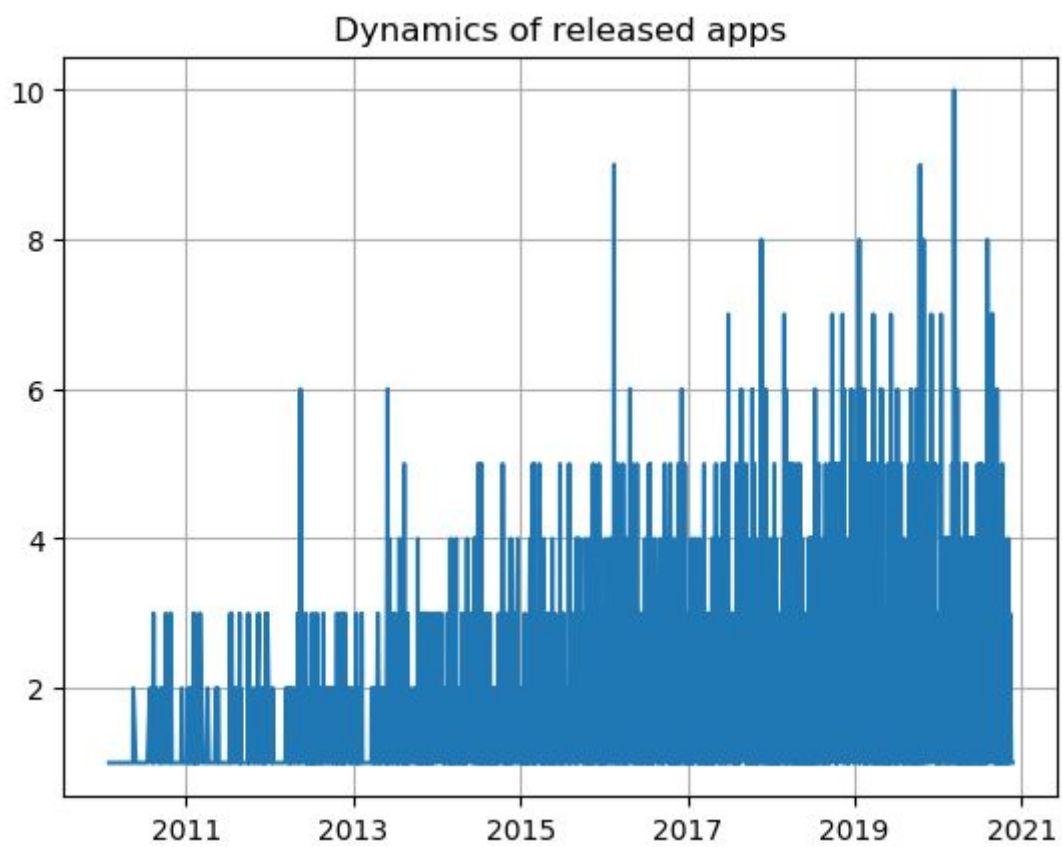
Рейтинг мобільних додатків



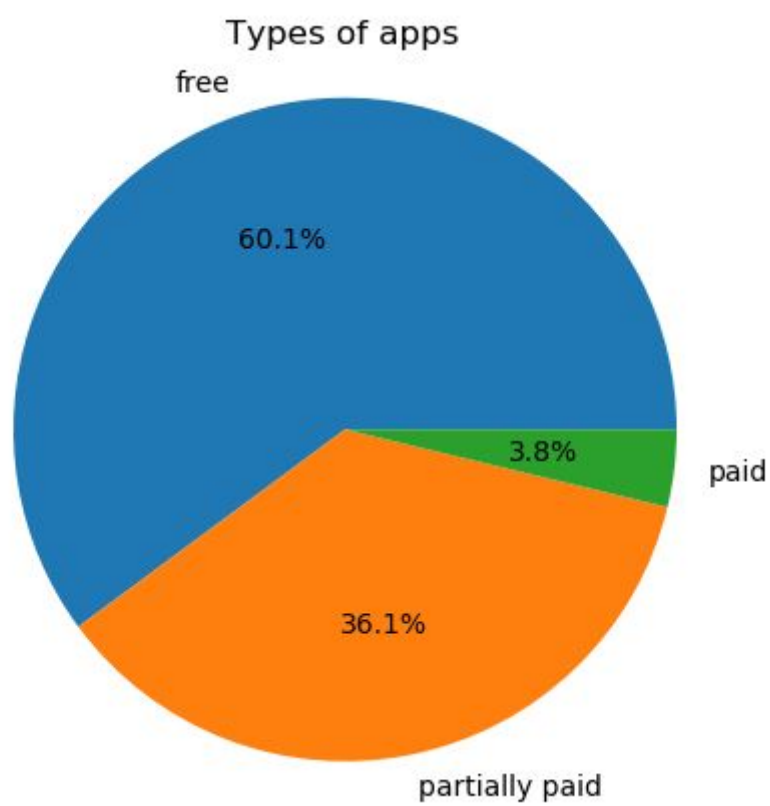
Рейтинг категорій



Рейтинг розробників



Динаміка релізів у Play Store



Типи додатків

Фрагменти програмного коду

Updating ts vectors using triggers

```
create trigger tsv_update_apps before insert or update
on apps for each row execute procedure
tsvector_update_trigger(app_name_tsv, 'pg_catalog.english', app_name);

create trigger tsv_update_comments before insert or update
on ratings for each row execute procedure
tsvector_update_trigger(comment_tsv, 'pg_catalog.english', comment);
```

```
def scrape_reviews(app_packages, quantity):
    app_reviews = []

    for ap in app_packages:
        temp, _ = reviews(
            ap,
            sort=Sort.NEWEST,
            count=quantity,
        )
        app_reviews.extend(temp)
        if len(app_reviews) >= quantity:
            break

    app_reviews_df = pd.DataFrame(app_reviews)
    app_reviews_df.to_csv('../data/csv/reviews.csv', header=True)

    return app_reviews
```

Використання модулю для скрепінгу

```
0 - exit
1 - entity menu
2 - graph analysis menu
3 - generation menu
4 - search menu
Enter number:
4
0 - go back
1 - search apps by similar title and category
2 - search developers by app rating and released date
3 - search users by similar comments
Enter number:
```

Консольне меню

```
def visualize_piechart(data, filename):
    names = []
    installs = []
    for d in data:
        names.append(d['name'])
        installs.append(d['installs'])
    sizes = percentage(installs)

    plt.figure()
    plt.pie(sizes, labels=names, autopct='%1.1f%%')
    plt.title('Types of apps')
    plt.axis('equal')
    plt.savefig('../visuals/' + filename + '.png', bbox_inches='tight')
    plt.show()
```

Одна з функцій візуалізації

```
def list_str(self, container, entity, action='Found', level=0):
    tabs = '\t' * level
    action_formatted = ' ' if len(action) == 0 else action + ' '
    result = ''
    if not container or container is None and action == 'Found':
        result += tabs + 'Not found\n'
    elif container is None:
        result += tabs + entity + 's is None\n'
    elif type(container) is not list:
        result += tabs + action_formatted + entity + ':\n' + self.dict_str(container, level + 1) + '\n'
    elif len(container) == 1:
        result += tabs + action_formatted + entity + ':\n' + self.dict_str(container[0], level + 1) + '\n'
    else:
        result += tabs + action_formatted + entity + 's:\n'
        for item in container:
            result += tabs + self.dict_str(item, level + 1) + '\n'
    print(result)
    return result
```

Вивід для сутностей